Chapter 6

Boolean Algebra

6.1 Introduction

Boolean algebra is the algebra of logic. It uses symbols to represent logical statements instead of words. Boolean algebra was formulated by the English Mathematician George Boole in 1847. Boolean algebra consists of rules for manipulating symbols. Boolean algebra has exactly the same structure as propositional calculus. The most important application of Boolean algebra is in digital logic. Computer chips are made up of transistors that are arranged in logical gates. Each gate performs a simple logical operation. The computer processes the logical propositions in its program by processing electrical pulses. The design of a particular circuit is based on a set of logical statements. These statements can be translated into the symbols of Boolean algebra. The algebraic statements can then be simplified according to the rules of the algebra, and translated into a simpler circuit design. Boolean algebra return results in terms of true or false i.e. 1 or 0 respectively.

Consider the following statements:

I am Pakistani	1
2 + 2 = 5	0
Lahore is the capital of Pakistan	0
5 + 1 = 6	1

Each of these statements is either **TRUE** or **FALSE**. Such statements are called **propositions.** The sentence **What is your name?** is not a proposition because it has no truth-value (TRUE or FALSE).

We can combine two propositions to form a new proposition as follows:

Let **p = Islamabad is the capital of Pakistan**

and **q = Sialkot is the capital of Punjab.**

Then **p** is **TRUE** and **q** is **FALSE**

Now form a new proposition **t** by using p and q as follows:

t = (Islamabad is the capital of Pakistan) **AND** (Sialkot is the capital of Punjab) or we may write that

t = p AND q

This proposition is **FALSE** because q is False and for **t** to be TRUE both p and q must be **TRUE.**

Similarly let **r = p OR q**

Clearly the proposition **r** is **TRUE** because **p** is **TRUE.**

Also with every proposition **p** we can make another proposition **q** as follows

Let **p = Islamabad is the capital of Pakistan**

Then make a new proposition **q** as follows:

q = NOT (Islamabad is the capital of Pakistan)

We may write **q = It is not TRUE that** Islamabad is the capital of Pakistan

q is called the negation of **p** and we write **q = NOT (p)** to express this idea.

It is obvious from the definition of negation given above that if p is TRUE then NOT (**p**) will be FALSE and if **p** is FALSE then NOT(**p**) will be TRUE.

Thus we have the following key points:

- Each proposition is either **TRUE** or **FALSE**
- We have two ways (AND ,OR) of combing two propositions to make new propositions
- Each proposition p has a negation NOT (p).

George Boole was actually interested in representing such a system of Logical sentences in a mathematical form.

Now let us consider another system, we know that all electronic devices consist of circuits of switches (Transistors). A switch at any given time is in one of the two states **ON or OFF**.

We can combine two switches A and B in the following two ways:

Series: The two switches A and B are arranged in a series as shown in the figure 6.1 the bulb will be **ON** if both switches are **ON** and it will be **OFF** otherwise.

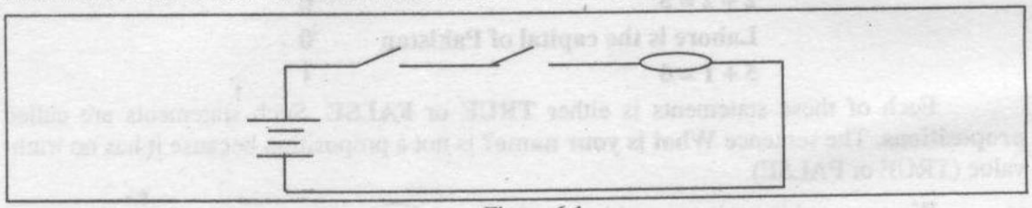

Figure 6.1

Parallel: If A and B are arranged in parallel as shown in the figure 6.2.

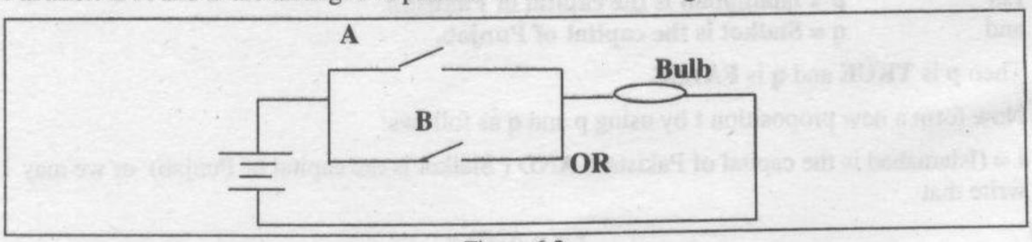

Figure 6.2

The bulb will be **ON** if atleast one of the switches is **ON** otherwise it will be **OFF**.

Serial circuit is represented by **.** operator and parallel circuit by + operator. This is explained as under:

Operations of AND (.)		
Switch A	Switch B	BULB
OFF	OFF	OFF
OFF	ON	OFF
ON	OFF	OFF
ON	ON	ON
Serial Circuit		

Operations of OR (+)		
Switch A	Switch B	BULB
OFF	OFF	OFF
OFF	ON	ON
ON	OFF	ON
ON	ON	ON
Parallel Circuit		

We can also represent the above circuits as expressions of the form **A. B** read as A dot B and **A+B** read as A plus B.

6.2 Boolean Algebra

Two valued Boolean algebra is a set that has two elements and two operations usually denoted by **.** and **+** are defined on the set such that the following axioms are satisfied

Close: Set *B* is closed under **.** and **+**

Commutative: Both the operations are commutative which means that if *a* and *b* are two variables that take values from the set, then

$$a + b = b + a \quad \text{and} \quad a \,.\, b = b \,.\, a$$

Associative: Both the operations are associative which means that if *a*, *b* and *c* are variables that take values from the set B, then

$$a+(b + c) = (a + b)+c \quad \text{and} \quad a.(b.c) = (a . b).c$$

Distributive: The **.** operation is distributive over + and + operation is distributive over **.** operation so if *a*, *b* and *c* are variables that take values from the set, then

$$a. (b + c) = (a . b) + (a . c) \quad \text{and} \quad a + (b. c) = (a + b).(a + c)$$

Identity: There is an identity element 1 with respect to **.** and an identity element 0 with respect to + such that for all *x*,

$$x. 1 = x \text{ and } x + 0 = x$$

Complement: Each element of the set B has a complement. If *x* is an element of set then its complement is denoted by $\bar{x}$ and has the following properties:

$$x + \bar{x} = 1 \quad \text{and} \quad x \,.\, \bar{x} = 0$$

Following example defines the most commonly used two valued Boolean algebra.

Example: Consider the set B = {0, 1} and two operations + and **.** on B as follows:

Operations of AND (.)		
x	*y*	*x . y*
0	0	0
0	1	0
1	0	0
1	1	1

Operations of OR (+)		
x	*y*	*x + y*
0	0	0
0	1	1
1	0	1
1	1	1

This set is a Boolean Algebra

Note that this addition operation is different from the usual addition because 1 + 1 = 1. Both the operations . and + have the following properties.

If *x, y* and *z* are variables that take values from the set B-then the set is closed under e + operator.

Close: Both $x \, . \, y$ and $x + y$ are members of B (i.e. $x \, . \, y$ and $x + y$ are either 0 or 1). Thus set B is closed under the operations . and +

Commutative: Following tables show that the operations of . and + commutative because from the tables it is obvious that

$$x \, . \, y = y \, . \, x \quad \textbf{and} \quad x + y = y + x$$

Tables to show that . and + are commutative.

x	y	$x \, . \, y$	$y \, . \, x$
0	0	0.0 = 0	0.0 = 0
0	1	0.1 = 0	1.0 = 0
1	0	1.0 = 0	0.1 = 0
1	1	1.1 = 1	1.1 = 1

x	y	$x + y$	$y + x$
0	0	0+0 = 0	0+0 = 0
0	1	0+1 = 1	1+0 = 1
1	0	1+0 = 1	0+1 = 1
1	1	1+1 = 1	1+1 = 1

Associative: Following table shows that for all values of the Boolean variables *x, y* and *z* we have $x \, . \, (y \, . \, z) = (x \, . \, y).z.$ so the AND operation is associative. Similarly we can show that OR operation is also associative i.e. $x + (y + z) = (x + y) + z.$

Table to show that . is associative

x	y	z	$x \, . \, y$	$(x \, . y) \, . z$	$y \, . \, z$	$x . (y \, . z)$
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	1	0
1	0	0	0	0	0	0
1	0	1	0	0	0	0
1	1	0	1	0	0	0
1	1	1	1	1	1	1

Distributive:

As is shown in the table below that for all possible values of *x, y* and *z*, we have

$$x \, . \, (y + z) = x \, . \, y + x \, . \, z$$

so is distributive over +. Similarly we can show that $x + (y \, . \, z) = (x + y) \, . \, (x + z)$

x	y	z	$x \, . \, y$	$x \, . \, z$	$x \, . y + x \, . \, z$	$y + z$	$x . (y + z)$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	0	1	0
0	1	1	0	0	0	1	0
1	0	0	0	0	0	0	0
1	0	1	0	1	1	1	1
1	1	0	1	0	1	1	1
1	1	1	1	1	1	1	1

Identity

As is obvious from the table given below that for any value of the variable x

$$0 + x = x \quad \text{and} \quad x \,.\, 1 = x$$

so 0 is the identity element of + and 1 is the identity element of **.**

x	$x\,.\,1$	$x + 0$
0	0	0
1	1	1

Complement

Every element of B has a complement. Complement of 0 is 1 and complement of 1 is 0 because

$$0 + 1 = 1 \quad \text{and} \quad 0\,.1 = 0$$

So the set B ={0,1} with the defined operations is a Boolean algebra because it satisfies all the axioms of Boolean Algebra.

Boolean Constants

If B = [0, 1] with operations **.** and + is a Boolean Algebra, then 0 and 1 are called Boolean constants.

Boolean Variables

If B = {0, 1} with operations **.** and + is a Boolean algebra, then the variables *x, y* etc are called Boolean variables. We can use the Boolean constants and variables to form Boolean expressions.

What are the Boolean constants in the Boolean algebra given in the examples?
What values can the Boolean variables take in Boolean Algebra?

Boolean Expressions

If *x, y* and *z* are Boolean variables and 0 and 1 are the Boolean constants, then by using the . , + and complement operations we can combine two or more variables and constants to make comparision.

$$x + y.\, z \quad \text{and} \quad \overline{x}\,.\,(y + z) \text{ etc.}$$

For evaluating a Boolean expression, we follow the following precedence of operations :

1. First of all evaluate all the complement operations
2. Secondly evaluate all the product **.**
3. Evaluate the addition operations + at the end

We can use parentheses to change the order of evaluation of operations in a Boolean expression. If parentheses are used, then first of all that part of expiession is evaluated which is within the parentheses.

Following example use these rules to evaluate different Boolean expressions:

Example 1: Evaluate $\overline{x}.y + x.\overline{z} + x.\overline{y}$ for $x = 0, y=1$ and $z = 0$

Solution:

First calculate complements as $x = 0$ so $\overline{x} = 1$ similarly $\overline{y} = 0$ and $\overline{z} = 1$

Now calculate products so $\overline{x}.y = 1.1 = 1$ $x.\overline{z} = 0.1 = 0$ and $x.\overline{y} = 0.1 = 0$

So $\overline{x}.y + x.\overline{z} + x.\overline{y} = 1+0+0 = 1$

Example 2: Evaluate $(x + y).\overline{x} + (\overline{y} + z)$ for $x = 0, y = 1$ and $z=1$

Solution:

First of all calculate comlements

$x + y = 0 + 1 = 1$ similarly $\overline{y} + z = 0 + 1$

Now $\overline{x} = 1$, $\overline{y} = 0$, $\overline{z} = 0$

So $(x + y).\overline{x} + (\overline{y} + z) = (0 + 1).1 + (0 + 1) = 1.1 + 1 = 1 + 1 = 1$

6.2.1 Evaluating an expression for all possible input values

Following examples shows the use of truth table for evaluating an expression for all possible input values.

Example 1: Evaluate the following Boolean expression. $x.\overline{y} + \overline{x}.y$ **using a truth table.**

Solution:

x	$\overline{x}$	y	$\overline{y}$	$x.\overline{y}$	$\overline{x}.y$	$x.\overline{y} + \overline{x}.y$
0	1	0	1	0	0	0 + 0 = 0
0	1	1	0	0	1	0 + 1 = 1
1	0	0	1	1	0	1 + 0 = 1
1	0	1	0	0	0	0 + 0 = 0

Example 2: Evaluate the following Boolean expression. $x.y + \overline{x}.y + y.\overline{z}$ using a truth table.

Solution:

x	$\overline{x}$	y	$\overline{y}$	z	$\overline{z}$	$x.y$	$\overline{x}.y$	$y.\overline{z}$	$x.y + \overline{x}.y + y.\overline{z}$
0	1	0	1	0	1	0	0	0	0
0	1	0	1	1	0	0	0	1	1
0	1	1	0	0	1	0	1	0	1
0	1	1	0	1	0	0	1	0	1
1	0	0	1	0	1	0	0	0	0
1	0	0	1	1	0	0	0	1	1
1	0	1	0	0	1	1	0	0	1
1	0	1	0	1	0	1	0	0	1

It is often very useful to construct a truth table of a given Boolean expression. It is important to note that truth table of a two variable expression will always have $2^2 = 4$ rows and truth table of a 3 variable expression will always have $2^3 = 8$ rows

6.2.2 Boolean Functions

Consider the Boolean expression $x + y$ where x and y are Boolean variables. Now let a function f as follows:

- f takes two Boolean constants as input
- f then calculate the value of the above expression at the input values
- The calculated value is the final answer of f.

Examples of two valued functions are:

$f(x, y) = x + y$ and $g(x, y) = \overline{x} . \overline{y} + x.y$

where x, y are Boolean variables.

Now consider another Boolean expression $x + y. z$, where x, y and z are Boolean variables. Now let us make the following rule for calculating the value of g as follows:

- g takes two Boolean constants as input
- g then calculate the value of the above expression at the input values
- The calculated value is the final answer of g.

Example: Represent the function $\mathbf{f}(x, y) = x. \overline{y} + \overline{x}.y$ by using a truth table

Solution:

x	y	$\overline{x}$	$\overline{y}$	$x. \overline{y}$	$\overline{x}.y$	$\mathbf{f}(x, y) = x. \overline{y} + \overline{x}.y$
0	0	1	1	0	0	0
0	1	1	0	0	1	1
1	0	0	1	1	0	1
1	1	0	0	0	0	0

This truth table shows the value of the functions for all the possible values of the parameters.

6.3 Laws and Theorems of Boolean Algebra

In this section we will see different laws of Boolean algebra and also prove some useful theorems. These theorems are used for simplifying different Boolean functions and in the simplification of different logical circuits

Theorem 1: If x is a Boolean variables then $x . x = x$ and $x + x = x$.This is also known as the idempotent law

We can prove this theorem in the following two ways:

- By using a truth table
- By using the axioms of Boolean algebra

Proof: By using a truth table

x	$x \cdot x$
0	$0 \cdot 0 = 0$
1	$1 \cdot 1 = 1$

From the truth table given above it is clear that if $x = 0$ then $x + x$ is also 0 and if $x = 1$ then $x + x$ is also 1 so we can say that $x + x = x$

Note: All theorems of Boolean algebra can be proved by using truth tables

by using the axioms of Boolean algebra

Now we shall prove the second part of this theorem by using the axioms of Boolean algebra and the definition of . and +as follows:

L.H.S $= x + x$

$= x.1 + x.1$ (by identity element)

$= x.(1+1)$ (by distributive law)

$= x.1$ $(1 + 1 = 1)$

$= x$ (by identity element)

= R.H.S

Hence the theorem is proved .

Note: The second part can be obtained by changing . into +
This fact will be very useful for proving certain theorems

Theorem 2: If x is a Boolean variable then $x + 1 = 1$ and $x \cdot 0 = 0$

We can prove this theorem using a truth table but that is left as an exercise for you. Here we shall prove this theorem by using axioms of Boolean algebra and previously proved theorems

Proof: L.H.S $= x + 1$

$= x + (x + \overline{x})$ (by definition of complement)

$= (x + x) + \overline{x}$ (by associative law)

$= x + \overline{x}$ (by idem potent law)

$= 1$ (by definition of complement)

= R.H.S

Note that:
We can use existing theorems to prove more theorems.

Now we shall prove the second part of this theorem that states $x.0 = 0$

L.H.S $= x \cdot 0$

$= x \cdot (x \cdot \overline{x})$ (by definition of complement)

$= (x \cdot x) \cdot \overline{x}$ (by associative law)

$= x \cdot \overline{x}$ (by idempotent law)

$= 0$ (by definition of complement)

=R.H.S

Hence the theorem is proved.

Theorem 3: For any Boolean variable x, $\overline{\overline{x}} = x$. This is also known as involution (or cancellation property).

Proof: As we stated earlier that every theorem can be proved by using a truth table. Here we will use a truth tale to prove this theorem

x	$\overline{x}$	$\overline{\overline{x}} = x$
0	1	0
1	0	1

The result can be obtained by comparing the first and third column of the truth table

Theorem 4: If x and y are Boolean variables then $x + \overline{x}.y = x + y$ and $x.(\overline{x} + y) = x$ this result is also known as the absorption law.

Proof: L.H.S $= x + x.y$

$= x.1 + x.y$ (1 is identity element)

$= x.(1 + y)$ (distributive law)

$= x(1)$ $(1 + y = 1)$

$= x$ (1 is the identity element)

= R.H.S

The proof of the second part is similar and is left for the students as an exercise.

Hence the result follows.

Theorem 5

De Morgan's law: The complement of addition of two numbers is equal to the product of their complements. Similarly the complement of product of two numbers is equal to the sum to their complements.

If x and y are two Boolean variables then

$\overline{x+y} = \overline{x}.\overline{y}$ and $\overline{x.y} = \overline{x} + \overline{y}$

Proof: We will prove the first result of this theorem by using truth table

x	y	$\overline{x}$	$\overline{y}$	$x+y$	$\overline{x+y}$	$\overline{x}.\overline{y}$
0	0	1	1	0	1	1
0	1	1	0	1	0	0
1	0	0	1	1	0	0
1	1	0	0	1	0	0

From the last two columns of this table it is obvious that $\overline{x+y} = \overline{x}.\overline{y}$

6.3.1 Duality Principle

The **Principle of Duality** states that any result deduced from the axioms of Boolean algebra remains valid if the following steps are performed

- All 0's in the result are changed to 1 and vice versa
- The . in the original result is changed to + and vice versa

Note: This result is very important because if we can prove a result of Boolean algebra then another valid result can be directly obtained from the proved result.

Example1: Prove that $\overline{x.y} = \overline{x} + \overline{y}$

Proof: We know from theorem 5 that $\overline{x+y} = \overline{x}.\overline{y}$ now applying the principle of duality on $\overline{x+y} = \overline{x}.\overline{y}$ gives us $\overline{x.y} = \overline{x} + \overline{y}$

Hence the result proved.

Example2: Apply the principle of duality to get the dual of the following expressions

$x.x = x$, $x+1=1$, $x + \overline{x}.y = x + y$, $\overline{x+y} = \overline{x}.\overline{y}$ and

$x.(y+z) = (x.y)+(x.z)$

Solution:

i. By changing the only . to + we get $x+x=x$

ii. By changing the + to . and changing 1 to 0 gives $x.0=0$

iii. By changing + to . and . to + gives $x.\overline{x}+y = x.y$

iv. Changing + to . and . to + gives us $\overline{x.y} = \overline{x} + \overline{y}$

v. Changing + to . and . to + gives us $x+(y.z) = (x+y).(x+z)$

6.3.2 Simplifying a Boolean function

It is clear from the above examples that every Boolean function can be represented as a combination of Boolean functions and also every circuit of logical gates can be represented as a Boolean expression. As the internal architecture of the computers memory and processor consists of these gates so it is always useful to find a simpler expression for representing a function. A simpler expression results into simple and efficient hardware.

In this section we shall learn the process of simplifying a given Boolean function. We will learn two ways of simplifying a Boolean function.

- Simplifying a Boolean function by using laws of Boolean algebra
- Simplifying a Boolean function by using k-map algorithm

The process of simplifying a Boolean function using laws of Boolean algebra are demonstrated in the following examples

Example 1: Simplify the following Boolean function f :

$$f(x, y) = x + \overline{x}.y$$

Solution:

$f(x, y) = x + \overline{x} . y$
$= (x + \overline{x}) . (x + y)$ (by Distributive law)
$= 1 . (x + y)$ (by Complement Definition)
$= (x + y)$ (by definition of identity element)

Clearly to implement the non-simplified function needs three logic gates whereas the implementation of the simplified function needs only 1 logic gate.

Example 2: Simplify the following Boolean function f :

$$f(x, y,z) = \overline{x}.y.z + x . \overline{y} + \overline{x} . \overline{y} .z$$

Solution:

$f(x, y,z) = \overline{x}.y.z + x . \overline{y} + \overline{x} . \overline{y} .z$
$= \overline{x}.y.z + \overline{x} . \overline{y} .z + x . \overline{y}$ (By commutative law)
$= \overline{x}.z (y + \overline{y}) + x . \overline{y}$ (By distributive law)
$= \overline{x}.z.1 + x.\overline{y}$ (By definition of Complements)
$= \overline{x}.z + x.\overline{y}$ (Identity element)

It is obvious that to implement the non-simplified function needs 9 logic gates whereas the implementation of the simplified function needs only 5 logic gates

Example 3: Simplify the following Boolean function f :

$$f(x, y, z) = x.z + \overline{x}.z.y$$

Solution:

$f(x, y, z) = x . z + \overline{x} . z . y$
$= x . z + \overline{x}.y . z$ (by associative and commutative law)
$= (x + \overline{x}.y). z$ (by Distributive law)
$= (x + y). z$ (by Idempotent law)
$= x . z + y.z$ (by Distributive law)

Clearly the simplified functions are much more desirable and useful.

6.3.3 Disadvantages of using Boolean Algebraic laws

Following is the list of disadvantages of using Boolean algebraic laws for simplification of Boolean expressions:

- It is very difficult to write a computer program (automate) that can use these laws to simplify a given Boolean function.
- This process may not give the best-simplified function and different people can have different simplified expressions.
- For this process to work a Boolean function is needed but in most engineering applications we do not have the actual Boolean function but have its truth table of the required function.

To overcome these disadvantages Maurice Karnaugh established another method for simplifying a Boolean expression. This method is based upon the Boolean algebraic laws but has non-of these disadvantages. This is commonly known as the k-map method of simplification.

Before we learn the next method of simplification let us learn the following terms.

Literals: If we have a Boolean function of two variables x and y then each variable can appear in the function in two forms i.e either the variable itself appears or it appears in the complement form. Each of these forms is called a literal. Each literal represent on input to the Boolean function.

6.3.4 Minterms (Standard Product)

If we have a two Boolean variables x and y then we can form the following four products using these variables. $x \cdot y$, $x \cdot \overline{y}$, $\overline{x} \cdot y$, $\overline{x} \cdot \overline{y}$. These are called standard products or minterms with two variables. Following example list all the minterms with three variables.

Example: List down all the minterms of three variables x, y, z. Also give a general formula for calculating the number of minterms with n variables.

Solution: With three variables we can form the following minterms

$x \cdot y \cdot z$, $x \cdot y \cdot \overline{z}$, $x \cdot \overline{y} \cdot z$, $x \cdot \overline{y} \cdot \overline{z}$,

$\overline{x} \cdot y \cdot z$, $\overline{x} \cdot y \cdot \overline{z}$, $\overline{x} \cdot \overline{y} \cdot z$, $\overline{x} \cdot \overline{y} \cdot \overline{z}$,

we can construct $4 = 2^2$ minterms with 2 variables and $8 = 2^3$ minterms with 3 variables. It is easy to see that we will have 2^n minterms with n variables. The table given below shows the way we name these minterms. It is important to remember the value of variables which is associated with a minterm.

Name	x	y	z	Minterm
m0	0	0	0	$\overline{x} \cdot \overline{y} \cdot \overline{z}$
m1	0	0	1	$\overline{x} \cdot \overline{y} \cdot z$
m2	0	1	0	$\overline{x} \cdot y \cdot \overline{z}$
m3	0	1	1	$\overline{x} \cdot y \cdot z$
m4	1	0	0	$x \cdot \overline{y} \cdot \overline{z}$
m5	1	0	1	$x \cdot \overline{y} \cdot z$
m6	1	1	0	$x \cdot y \cdot \overline{z}$
m7	1	1	1	$x \cdot y \cdot z$

Table of Names of minterms

6.3.5 Maxterms (Standard Sum)

If we have a two Boolean variables x and y, then we can form the following four sums using these variables. $x + y$, $x + \overline{y}$, $\overline{x} + y$, $\overline{x} + \overline{y}$. These are called a standard sums or maxterms with two variables. It is easy to see that we will have 2^n maxterms with n Boolean variables. The table given below shows the way we name these Maxterms.

Name	x	y	z	Maxterms
M0	0	0	0	$x + y + z$
M1	0	0	1	$x + y + \overline{z}$
M2	0	1	0	$x + \overline{y} + z$
M3	0	1	1	$x + \overline{y} + \overline{z}$
M4	1	0	0	$\overline{x} + y + z$
M5	1	0	1	$\overline{x} + y + \overline{z}$
M6	1	1	0	$\overline{x} + \overline{y} + z$
M7	1	1	1	$\overline{x} + \overline{y} + \overline{z}$

Table of Names of Maxterms

The concept of minters and maxterms is very useful for simplifying a Boolean function to a minimum number of literals.

Another important idea is that we can write every Boolean function as Sum of minterms or as Product of Maxterms. We will learn the Minterms concept in detail and leave the Maxterms for the next classes.

6.4 Karnaugh Map (K-Map)

Karnaugh Map is a very efficient way of solving Boolean functions. In this section we will learn to solve a two and three variables Boolean function in the form of a map.

6.4.1 Map for a two variable Boolean function

Following figure shows the arrangement of a two variable Boolean function in the form of a map so the square in row 0 and column 0 is mo and for the minterm the square at row 0 column 1 is m1.

x \ y	0	1
0	m0	m1
1	m2	m3

Let us Consider the function as sum of minterms as follows $f(x,y,z) = \overline{x} \cdot y + x \cdot \overline{y}$. This function can be written in a k-map as follows

x \ y	$\overline{y}$	y
$\overline{x}$	0	1
x	0	0

So to express a function in the form of a k-map we determine the minterms in that function and then write 1 in all those squares which correspond to a minterm present in the function and write 0 in the remaining squares.

6.4.2 Map for a three variable Boolean function

The map for representing a three variable function is shown below:

	x\ y.z	0.0 ($\overline{y}.\overline{z}$)	0.1 ($\overline{y}.z$)	1.1 ($y.z$)	1.0 ($y.\overline{z}$)
$\overline{x}$	0	m0	m1	m3	m2
x	1	m4	m5	m7	m6

It is extremely important to arrange the rows and columns as given in the above table. The process of representing a three valued function in a k-map is the same as for the two variable functions. Following examples show the process of representing a Boolean function in the form of k-map.

Example 1: Represent the following Boolean function in a three variable k-map

$$f(x, y, z) = x \cdot y \cdot \overline{z} + \overline{x} \cdot \overline{y} \cdot \overline{z} + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot y \cdot z$$

Solution:

Step 1: First represent the function as sum of minterms form.

$$f(x, y, z) = x \cdot y \cdot \overline{z} + \overline{x} \cdot \overline{y} \cdot \overline{z} + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot y \cdot z$$

This function is already in the required form

Step 2: For each minterm present in the function Mark a 1 in the corresponding square in the map and mark a 0 in all other squares

x\y.z	0.0	0.1	1.1	1.0
	$\overline{y} \cdot \overline{z}$	$\overline{y} \cdot z$	$y \cdot z$	$y \cdot \overline{z}$
0 $\overline{x}$	1	0	1	0
1 x	1	0	0	1

Example 2: Represent the following Boolean function in a two variable k-map

$$f(x, y) = y$$

Solution:

Step 1: First represent the function as sum of minterms form

$$\begin{aligned} f(x, y) &= y \\ &= (x + \overline{x}) \cdot y \\ &= x \cdot y + \overline{x} \cdot y \end{aligned}$$

Step 2: For each minterm present in the function Mark a 1 in the corresponding square in the map

x\y		0	1
		$\overline{y}$	y
0	$\overline{x}$	0	1
1	x	0	1

6.4.3 Simplifying a Boolean Function of Two Variables Using k-map

Following examples show the process of simplification of a two variable Boolean function using a k-map.

Example 1: Simplify the Boolean function $f(x, y) = x . y + \overline{x} . y$

Solution:

Step 1: Represent the function in the form of a k-map. This is shown below:

$x \backslash y$	$\overline{y}$	y
$\overline{x}$	0	1
x	0	1

Step 2: Mark any groups of two or four adjacent 1 as shown below

$x \backslash y$	$\overline{y}$	y
$\overline{x}$	0	1
x	0	1

Step 3: Write simplified expression for each group.

The grouped minterms are $\overline{x} . y$ and $x . y$ as the value of x changes so we can write the following expression for this group of minterms

$$\overline{x} . y + x . y = y$$

Step 4: Write the final simplified form as a sum of products

$$f(x, y) = y$$

Example 2: Simplify the Boolean function $f(x, y) = x . \overline{y} + \overline{x} . y + x . y$

Solution:

Step 1: Represent the function in the form of a k-map. This is shown below

	$x \backslash y$	**0**	**1**
$\overline{x}$	**0**	0	1
x	**1**	1	1

Step 2: Mark any groups of two or four adjacent 1 as shown below.

	$x \backslash y$	**0**	**1**
$\overline{x}$	**0**	0	1
x	**1**	1	1

Step 3: Write simplified expression for each group.

The grouped minterms are $\overline{x} . y$ and $x . y$ and another group of minterms is $x . \overline{y}$ and $x . y$. As the value of x changes in the first group and value of y changes in the second group.

so expression for the first group = y

so expression for the second group = x

Step 4: Write the final simplified form as a sum of products

$$f(x, y) = x + y.$$

Example 3: Simplify the Boolean function

$$f(x, y) = x.\ \overline{y} + \overline{x}.y + x.y + \overline{x}\ .\ \overline{y}$$

Solution:

Step 1: Represent the function in the form of a k-map. This is shown below

$x \backslash y$	$\overline{y}0$	$y1$
0	1	1
1	1	1

Step 2: Mark any groups of two or four adjacent 1 as shown below:

$x \backslash y$	**0**	**1**
$\overline{x}$ **0**	1	1
x **1**	1	1

Step 3: Write simplified expression for each group. All the elements are 1 and there is only one group .

Step 4: Write the final simplified form as a sum of products in this case we write, **$f(x, y) = 1$** because it is always 1

Example 4: Simplify the Boolean function $f(x, y) = x.\ \overline{y} + \overline{x}.y$

Solution:

Step 1: Represent the function in the form of a k-map. This is shown below

$x \backslash y$	$\overline{y}0$	$y1$
0	0	1
1	1	0

Step 2: Mark any groups of two or four adjacent 1 as shown below :

$x \backslash y$	$\overline{y}\,0$	$y1$
$\overline{x}$ **0**	0	1
x **1**	1	0

Note that the elements along the diagonal are not adjacent to each other

Step 3: Write simplified expression for each group

as there are no groups so we write the minterm corresponding to the each 1 in the map

$$x.\ \overline{y} \text{ and } \overline{x}.y$$

Step 4: Write the final simplified form as a sum of products

$$f(x, y) = x.\overline{y} + \overline{x}.y$$

6.4.4 Simplifying a Boolean Function of Three Variables Using k-map

Following examples show the process of simplification of a three variable Boolean function using k-map.

Example 1: Simplify the Boolean function

$f(x, y, z) = x.y.\overline{z} + \overline{x}.\overline{y}.\overline{z} + x.\overline{y}.\overline{z} + \overline{x}.y.z$

Solution:

Step 1: Represent the function in the form of a k-map. This is shown below

$x \backslash y.z$	$\overline{y}.\overline{z}$	$\overline{y}.z$	$y.z$	$y.\overline{z}$
$\overline{x}$	1	0	1	0
x	1	0	0	1

Step 2: Mark any groups of two or four adjacent 1 as shown below

$x \backslash y.z$	$\bar{y}.\bar{z}$	$\bar{y}.z$	$y.z$	$y.\bar{z}$
$\bar{x}$	1	0	1	0
x	1	0	0	1

Group 1: $\bar{x}.\bar{y}.\bar{z}$ and $x.\bar{y}.\bar{z}$ first row

Group 2: $x.\bar{y}.\bar{z}$ and $x.y.\bar{z}$ so third column

Ungrouped terms: $\bar{x}.y.z$

Step 3: Write simplified expression for each group.
as there are two groups so we write the minterm corresponding to the each 1 in the map

Group 1: $\bar{x}.\bar{y}.\bar{z}$ and $x.\bar{y}.\bar{z}$ so simplified expression is $\bar{y}.\bar{z}$ (x will vanish)

Group 2: $x.\bar{y}.\bar{z}$ and $x.y.\bar{z}$ so simplified expression is $x.\bar{z}$ (y will vanish)

Step 4: Write the final simplified form as a sum of products, the ungrouped term will be added as it is

$$f(x, y, z) = \bar{y}.\bar{z} + x.\bar{z} + \bar{x}.y.z$$

Example 2: Simplify the Boolean function

$$f(x, y) = \bar{x}.y.z + \bar{x}.y.\bar{z} + x.y.z + x.y.\bar{z} + x.\bar{y}.\bar{z}$$

Solution:

Step 1: Represent the function in the form of a k-map. This is shown below

$x \backslash y.z$	$\bar{y}.\bar{z}$	$\bar{y}.z$	$y.z$	$y.\bar{z}$
$\bar{x}$	0	0	1	1
x	1	0	1	1

Step 2: Mark any groups of two or four adjacent 1 as shown below

$x \backslash y.z$	$\bar{y}.\bar{z}$	$\bar{y}.z$	$y.z$	$y.\bar{z}$
$\bar{x}$	0	0	1	1
x	1	0	1	1

The groups are

Group 1: $\bar{x}.y.z$, $x.y.z$, $\bar{x}.y.\bar{z}$, $\bar{x}.y.\bar{z}$

Group 2: $x.\bar{y}.\bar{z}$ $x.y.\bar{z}$

It is extremely important to note that the squares on the left edge are taken to be adjacent to the squares on the right edge. These form the group 2 and have been marked by using rectangular shape

Step 3: Write simplified expression for each group.

The simplified form of group 1 is y because both x, $\overline{x}$, and z, $\overline{z}$ are changing in the group.

Also the simplified form of group 2 is $x \cdot \overline{z}$ because both y and $\overline{y}$ are changing in the group.

Step 4: Write the final simplified form as a sum of products

$$f(x, y) = y + x \cdot \overline{z}$$

Example 2: Simplify the Boolean function

$f(x, y) = x.y.\overline{z} + \overline{x}.\overline{y}.\overline{z} + x.\overline{y}.\overline{z} + \overline{x}.y.z + \overline{x}.\overline{y}.z + \overline{x}.y.\overline{z} + x.y.z$

Solution:

Step 1: Represent the function in the form of a k-map. This is shown below

$x\backslash y.z$	$\overline{y}.\overline{z}$	$\overline{y}.z$	$y.z$	$y.\overline{z}$
$\overline{x}$	1	1	1	1
x	1	0	1	1

Step 2: Mark any groups of two or four adjacent 1 as shown below

$x\backslash y.z$	$\overline{y}.\overline{z}$	$\overline{y}.z$	$y.z$	$y.\overline{z}$
$\overline{x}$	1	1	1	1
x	1	0	1	1

so there are three groups

Group 1: $\overline{x}.\overline{y}.\overline{z}$ $\quad \overline{x}.\overline{y}.z$ $\quad \overline{x}.y.z$ $\quad \overline{x}.y.\overline{z}$ (Top Row)

Group 2: $\overline{x}.y.z$ $\quad \overline{x}.y.\overline{z}$ $\quad x.y.z$ $\quad x.y.\overline{z}$ (Last two columns)

Groupt 3: $\overline{x}.\overline{y}.\overline{z}$ $\quad x.\overline{y}.\overline{z}$ $\quad \overline{x}.y.\overline{z}$ $\quad x.y.\overline{z}$ (First and last column)

Once again note that the squares on the left edge are taken to be adjacent to the squares on the right edge. These form the group 2 and have been marked by using rectangular shape. Also note that a minter can be used in more than one group.

Step 3: Write simplified expression for each group.

Group1: becomes $\overline{x}$

Group2: becomes y

Group3: becomes $\overline{z}$

Step 4: Write the final simplified form as a sum of products

$$f(x, y, z) = \overline{x} + y + \overline{z}$$

You can also notice that a group of two 1's eliminates one literal, a group of four 1's eliminates two literals and a group of eight 1's eliminates three literals. So if all the squares have 1's then all literals are eliminated and function becomes constant i.e., 1

Advantages and Disadvantages of k-map method

Some advantages of this method of simplification are given below.

- This method is very easy to follow.
- This is a systematic process. It always leads to a single minimal solution.

A disadvantage of this system is that it is not scalable. This means that this system works very well for less variables but becomes complex for higher number of variables

Exercise 6

1. State and prove the **De Morgan's laws** for the Boolean algebra.
2. If x and y are Boolean variables then prove the following identities by using truth table.
   - **a.** $\overline{x} + \overline{y}$
   - **b.** $x + x.y = x + y$
   - **c.** $x.(x + y) = x$
   - **d.** $x + 1 = 1$
   - **e.** $x.0 = 0$
3. Make truth table of the following functions:
   - **a.** $f(x, y) = x.y + \overline{x}.y$
   - **b.** $x.\overline{y} + \overline{x}.y$
4. Calculate the value of the following Boolean functions at the given values of x, y and z.
   - **a.** $\overline{x}.y + \overline{x}.\overline{z} + x.\overline{y}$ for $x = 0, y = 1$ and $z = 0$
   - **b.** $(\overline{x} + y).x + (\overline{y} + z)$ for $x = 0, y = 1$ and $z = 1$
5. Prove the following results and apply the principle of duality to obtain the dual of these results.
   - **a.** $x + \overline{x} = x$
   - **b.** $x + 0 = x$
   - **c.** $\overline{x} + x.y = \overline{x} + y$
   - **d.** $\overline{x}.(y + z) = (\overline{x}.y) + (\overline{x}.z)$
6. Explain the following Logic gates and show their function by using a truth table.
   - **a.** AND
   - **b.** OR
   - **c.** NOT
7. Represent the following Boolean expressions as a combination of logic gates.
   - **a.** $x.\overline{y} + \overline{x}.y$
   - **b.** $x.\overline{y} + \overline{x}.y$
   - **c.** $\overline{x} + \overline{x}.y$
8. Simplify the following Boolean functions using K-maps:
   - **a.** $f(x, y) = x + \overline{x}.y$
   - **b.** $f(x, y, z) = \overline{x}.y.z + x.\overline{y} + \overline{x}.\overline{y}.z$
   - **c.** $f(x, y, z) = x.z + \overline{x}.z.y$
9. Fill in the blanks.
   - **(i)** Commutative laws states that $a + b$ is equal to_____________.
   - **(ii)** By distributive law we know that $ab + ac$ is equal to _____________.
   - **(iii)** A+ 0 is equal to_____________.
   - **(iv)** 0 is called the _____________.
   - **(v)** Boolean Algebra operates on _____________.

(vi) In Boolean algebra the identity element with respect to dot (.) is ________
(vii) $x + x$ is equal to ________
(viii) ________ is a very efficient way of solving Boolean functions.
(ix) $\overline{x \cdot y}$ is equal to ________
(x) In Boolean algebra standard product is called ________.

10. Match the following

$(a + b)$	Sum of products
Minterms	$x.0=0$
Maxterms	Product of Sums
$x+1=1$	$a.b$

11. Choose the correct answer.

(i) K-Map is used to
a. Evaluate a Boolean expression **b.** Simplify a Boolean expression
c. Both a and b **d.** Non of above

(ii) Demorgan's Law states that
a. $a(b+c) = a.b + a.c$ **b.** $a + (b+c) = (a+b) + c$ **c.** $\overline{a+b} = \overline{a} . \overline{b}$ **d.** none of above

(iii) A Boolean function with four variables will have
a. 8 maxterms **b.** 16 maxterms **c.** 24 maxterms **d.** 32 maxterms

(iv) The idempotent law states that for two variables x and y
a. $x + x . y = x + y$ and $x . (x + y) = x$ **b.** $\overline{\overline{x}} = x$ **c.** $x.x = x$ and $x + x = x$
d. none of the above

(v) The absorption law states that for two variables x and y
a. $x.x = x$ and $y.y = y$ **b.** $x. \overline{y} = \overline{y}.x$ **c.** $x + x . y = x + y$ and $x . (x + y) = x$
d. none of the above

12. Mark the following as True/False.

(i) Idempotent law states that $x+1 = 1$.
(ii) K-map is used to simplify a Boolean expression . **(iii)** $x + y + z$ is a minterm.
(iv) k-map may or may not lead to a single minimal solution .
(v) A Boolean function cannot involve more than two variables .
(vi) The principle of duality states that . and + are interchangeable .
(vii) As the number of variables in a Boolean function increase, the k-map becomes more complex .
(viii) A Boolean function involving 5 variables will have 31 minterms .
(ix) To simplify a k-map of groups of two, four, six, or eight 1s can be marked .
(x) Involution principle states that $y + \overline{y} = 1$.

Answers

Q.9

(i) $b + a$	(ii) $a . (b + c)$	(iii) A	(iv) additive identity	(v) Binary numbers
(vi) 1	(vii) x	(viii) K-map	(ix) $\overline{x} + \overline{y}$	(x) minterm

Q.11

(i) c	(ii) c	(iii) b	(iv) c	(v) c

Q.12

(i) F	(ii) T	(iii) F	(iv) F	(v) F
(vi) T	(vii) T	(viii) F	(ix) F	(x) F